**LAB 3 “MySQL”**

**( Ahmed Ashraf Ibrahim )**

MySQL Labs

**insert into students\_courses**

**values**

**(1,4,60,NULL), (2,1,NULL,NULL), (2,4,75,NULL), (3,1,NULL,NULL), (3,2,NULL,NULL), (3,3,75,NULL);**

|  |  |
| --- | --- |
| *1* | ***Create function to calculate the number of students who get grade less than 80 in a certain exam (course id will be sent as a parameter)*** |
|  | **drop function if exists num\_of\_students;**  **delimiter #**  **CREATE FUNCTION num\_of\_students(p\_id integer)**  **RETURNS int(11)**  **BEGIN**  **Declare v\_count integer;**  **SET v\_count =**  **(select count(student\_id) from students\_courses**  **where course\_id = p\_id and grade < 80) ;**  **RETURN v\_count;**  **END#**  **delimiter ;** |
| *2* | ***Create stored procedure to display the names of the absence students of a certain courses.(Absent means has no grades)*** |
|  | **drop procedure if exists names\_of\_absence;**  **delimiter #**  **CREATE PROCEDURE names\_of\_absence(p\_id integer)**  **BEGIN**  **SELECT concat (first\_name ,' ', last\_name) as Full\_Name**  **from students s , students\_courses s\_c**  **where course\_id = p\_id**  **AND s.student\_id = s\_c.student\_id**  **AND s\_c.grade Is NULL ;**  **END#**  **delimiter ;**    **CALL names\_of\_absence(1) ;** |
| *3* | ***Create stored procedure to calculate the average grades for certain course.*** |
|  | **Drop procedure if exists average\_grades ;**  **Delimiter #**  **Create procedure average\_grades(p\_id integer)**  **BEGIN**  **Select c.course\_name, avg(s\_c.grade)**  **From courses c, students\_courses s\_c**  **WHERE c.course\_id = s\_c.course\_id**  **AND s\_c.course\_id = p\_id**  **Group by c.course\_name;**  **END#**  **Delimiter ;** |
| *4* | ***Create trigger to keep track the changes(updates) of the grades in the studnets\_courses table***  ***( create changes table with the following fields:***  ***id int primary key auto\_increment , user varchar(30) , action varchar(40),***  ***old\_grade int, new\_grade int, change\_date date).***  ***Test the trigger by updating grade int the “Students\_courses” table***  ***Confirm that the row is added in the” change\_table”*** |
|  | **create table if not exists changes(id int primary key auto\_increment ,**  **-> user varchar(30) , action varchar(40), old\_grade int,**  **-> new\_grade int, change\_date date) ;**    **> delimiter #**  **> CREATE TRIGGER update\_grade**  **-> AFTER Update ON students\_courses**  **-> FOR EACH ROW**  **-> INSERT INTO changes(user, action, old\_grade,**  **new\_grade, change\_date)**    **-> VALUES (current\_user(), "update" , OLD.grade,**  **NEW.grade , current\_date());**  **-> END#**  **> delimiter ;**    **update students\_courses**  **-> set grade = 100**  **-> where student\_id = 1**    **-> And course\_id = 1 ;** |
| *5* | ***Create event to delete the changes tables every 5 minute*** |
|  | **set @@global.event\_scheduler =1;**  **select @@global.event\_scheduler;**    **CREATE EVENT delete\_changes**  **ON SCHEDULE EVERY 5 MINUTES**  **DO**  **DELETE FROM changes** |